Memento pattern program

Program description

The program allows the user to edit and save an employee’s data which consists of the employee’s first and last names, job, age and address. Additionally, the program implements the memento pattern and allows the user to create mementos of the employee’s data at a certain point in time (separate from the function to save the employee’s data) and allows the user to revert the employee’s data back to any of these mementos by selecting them from a list box based on the time at which they were created.

Object description

1. Form1 – main form of the program, which also acts as the “caretaker” for the memento pattern

* currentEmployee – variable of type *Employee* used to store the employee whose data the program is currently using
* employeeMementos – a list of mementos (*EmployeeMemento* objects) related to *currentEmployee*
* DisplayDetails() – a method to display/update the current employee’s (*currentEmployee*) data in the form
* btnNewMemento\_Click() – event method used when the “New memento” button is clicked; saves the current employee’s details from the form’s textboxes, creates a new memento instance and adds it to *employeeMementos*, and then displays the new memento’s creation time in the “Memento creation time” list box
* btnSave\_Click() – event method used when the “Save changes” button is clicked; Displays a notification if the employee’s data hasn’t been changed and there’s nothing to save, otherwise saves the changes made to the employee’s data
* btnRevert\_Click() – if a memento is selected from the “Memento creation time” list box, reverts *currentEmployee* to the memento and updates the form’s details; Otherwise notifies the user that no memento is selected

1. Employee – class used to store an employee’s data

* Employee() – standard, empty class constructor
* Employee(string fName, string lName, string add, string pos, int mori) – class constructor that initializes the employee’s information upon initialization
* String firstName – private variable to store the employee’s first name
* String FirstName – property to access the *firstName* variable
* String lastName – private variable to store the employee’s last name
* String LastName – property to access the *lastName* variable
* String address – private variable to store the employee’s address
* String Address – property to access the *address* variable
* String job – private variable to store the employee’s job
* String Job – property to access the *job* variable
* Int age – private variable to store the employee’s age
* Int Age – property to access the *age* variable

1. EmployeeMemento – memento class used to store a snapshot of an *Employee* class’ data at a certain point in time

* EmployeeMemento(Employee main) – class constructor that takes in an *Employee* object and stores its data, along with the time at which the *EmployeeMemento* was initialized
* String created – private variable to store the time at which the class instance was initialized
* String Created – read-only property to access the *created* variable
* String firstName – private variable to store the employee’s first name, corresponds to the variable with the same name in the *Employee* class but has no property to access it outside the *EmployeeMemento* class
* String lastName – private variable to store the employee’s last name, corresponds to the variable with the same name in the *Employee* class but has no property to access it outside the *EmployeeMemento* class
* String address – private variable to store the employee’s address, corresponds to the variable with the same name in the *Employee* class but has no property to access it outside the *EmployeeMemento* class
* String job – private variable to store the employee’s job, corresponds to the variable with the same name in the *Employee* class but has no property to access it outside the *EmployeeMemento* class
* Int age – private variable to store the employee’s age, corresponds to the variable with the same name in the *Employee* class but has no property to access it outside the *EmployeeMemento* class
* GetMemento() – constructs a temporary *Employee* object using the employee data stored in the *EmployeeMemento* class and then returns the *Employee* object to the caller

3 problems

Reusability – The program can easily be reused to manipulate employee (or any other sort of entity) data and track changes made, for example in a personnel management system.

Maintainability – The memento pattern implements comparatively short, clear and simple code, so maintaining it should not be an issue.

Extendibility – The program can be easily extended, so as to for example allow for multiple employees’ data to be saved (along with their respective mementos) or, due to the short and simple nature of the code, include additional employee information.